Amendments to the Claims:

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Original) A system for processing multi-agent cooperative transactions comprising:

a) a failure detector for detecting whether a failure is an inter-enterprise failure or an intra-enterprise failure;

b) an intra-enterprise failure handler coupled to the failure detector for performing failure recovery for intra-enterprise failures; and

c) an inter-enterprise failure handler coupled to the failure detector for performing failure recovery for inter-enterprise failures.

2. (Original) The system of claim 1 wherein the intra-enterprise failure handler includes

a scope determination module for identifying the failure recovery scope; and

a top-down logical undo module coupled to the scope determination module for undoing sub-transactions in the identified scope in a top-down manner.

3. (Original) The system of claim 2 wherein the scope determination module terminates at a closest ancestor of the failed transaction that is one of non-vital and associated with a contingency transaction.

4. (Original) The system of claim 1 wherein the scope determination module

when the failed transaction is non-vital to is parent, continuing with the parent transaction;

when is determined to be a result of agent malfunction, the parent transaction delegating the sub-transaction to another agent; and

when the failed transaction is associated with a contingency transaction, re-trying the contingency transaction.

5. (Original) The system of claim 1 wherein the inter-enterprise failure handler includes

a scope determination module for identifying the failure recovery scope; and

a top-down logical undo module coupled to the scope determination module for undoing sub-transactions in the identified scope in a top-down manner.

6. (Original) The system of claim 5 wherein the top-down logical undo module

when the transferred in transaction has not started, then not performing failure handling on the transferred in transaction;

when the transferred in transaction is in progress, then determining the rollback root of the transferred in transaction; and

when the transferred in transaction has completed, then compensating for the transferred in transaction.

7. (Original) The system of claim 1 wherein the top-down logical undo module

when the rollback root of the transferred in transaction, then utilizing the rollback root as the root of recovery;

when the rollback root of the transferred in transaction has committed to one of a parent and an ancestor, the parent is in-progress, and the rollback root of the transferred in transaction is one of non-vital and associated with a contingency transaction, then utilizing the parent of the rollback root as the root of recovery; and

when the rollback root of the transferred in transaction and the parent of the rollback root have committed, then determining the highest committed ancestor of the rollback root of the transferred in

transaction and determining whether the rollback root of the highest committed ancestor is in progress;

when the highest committed ancestor is in progress, the highest committed ancestor of the rollback root of the transferred in transaction is utilized as an extended rollback root;

when the highest committed ancestor is not in progress, the parent of the highest committed ancestor of the rollback root of the transferred in transaction is utilized as an extended rollback root.

8. (Withdrawn) A method of processing multi-agent cooperative transactions comprising the steps of:

a) a first transaction waiting for a commit notification; the first transaction including a dependency on a second transaction;

b) the second transaction sending a commit notification; and

c) the first transaction receiving the commit notification and committing only after receipt of the commit notification.

9. (Withdrawn) The method of claim 8 wherein the dependency is a start dependency; and wherein the first transaction does not initiate processing until the second transaction is initiated.

10. (Withdrawn) The method of claim 8 wherein the dependency is a failure dependency; wherein the first transaction aborts when the second transaction aborts.

11. (Withdrawn) The method of claim 8 wherein the dependency between the first transaction and a second transaction is a mutual settle dependency; and wherein the first transaction and the second transaction can commit only when the first transaction and the second transaction have reached agreement on a contract.

12. (Withdrawn) The method of claim 8 wherein the dependency between the first transaction and a second transaction is a mutual abort-compensate dependency; and wherein when a first transaction falls, then the second transaction is either aborted or compensated for.

13. (Withdrawn) The method of claim 8 wherein sending a commit notification employs a point-to-point communication.

14. (Withdrawn) The method of claim 8 wherein sending a commit notification employs a multicast communication.

15. (Withdrawn) The method of claim 8 wherein the step of the first transaction receiving the commit notification and committing only after receipt of the commit notification includes

committing to a first database associated with the first transaction.

16. (Withdrawn) The method of claim 8 wherein a centralized coordinator for controlling the cooperation between transaction is not needed.

17. (Original) A method for processing failures in multi-agent cooperative transactions comprising:

a) detecting whether a failure is an inter-enterprise failure or an intra-enterprise failure;

b) when the failure is an intra-enterprise failure, performing failure recovery for the intra-enterprise failure; and

c) when the failure is an inter-enterprise failure, performing failure recovery for the inter-enterprise failure.

18. (Original) The method of claim 17 wherein the step of when the failure is an intra-enterprise failure, performing failure recovery for the intra-enterprise failure includes

identifying the failure recovery scope; and

undoing sub-transactions in the identified scope in a top-down manner.

19. (Original) The method of claim 17 wherein the step of when the failure is an inter-enterprise failure, performing failure recovery for the inter-enterprise failure includes

identifying the failure recovery scope; and

undoing sub-transactions in the identified scope in a top-down manner.

20. (Original) The method of claim 19 wherein identifying the failure recovery scope includes the step of terminating at a closest ancestor of the failed transaction that is one of non-vital and associated with a contingency transaction.